

Computer Users' Group
January, 1985 Vol 7 No 5

Those of us who were here last year (83-84) remember very well that the computer went down for 3 days starting Christmas Eve. Well it happened again this Christmas. It makes you wonder if the Computer Centre, or CDC for that matter, added a special switch that shuts down the computer on Christmas Eve so that everybody will enjoy those 3 days , and so no one will slave over a hot computer... Anyway the bug was trapped on the morning of Thursday, Dec. 27, and everything was back to normal.

If you've used the PC network before, you will be very surprised by the changes implemented over Christmas.

course, they will be re-installed.

Also, the logical names for the disk drives have been changed; these used to be L:, G: and P: for the Library, Group and Personal drives, respectively. Rumour has it that certain software on the network wouldn't run with names as high as these (since on most systems a drive name of P: indicates that there are 16 physical drives available on one computer!). In any case, the library drive is now known as drive D:, the group drive is drive G:, and the personal drive is now E:. Secondly, the print utility was changed; PTEST is no more, and instead we have something called QPRINT, which works identically to PTEST.

Finally, a word of caution. For those of you who like to use the PATH statement to make life a little easier fetching stuff from the G: or D: drives while working on E:, you are going to have to make a slight adjustment. It seems that some of the new network software assumes that there is a "PATH C:\" statement in effect. If you change this, it makes it very difficult to do anything at all, such as log off, get back into the menu or print files. The way to get around it is to use "PATH C:\;G:\;D:\".

Or, Everything You Ever Wanted
to Know About Timesharing but Were
Afraid to Ask.

Want to learn timesharing? Run
your programs without having to send
them to the printer? Impress your

friends?

CUG to the rescue! We've prepared a manual on timesharing written especially for students. Don't let the word 'manual' scare you. This is an easy-to-read, understandable, entertaining and informative document in which you'll learn the basics like logging on and off of timesharing terminals, creating files and the 'recipes' necessary for running programs in your favorite language be it Pascal, COBOL, FORTRAN or BASIC! You can learn how to use various packages such as SPSS, MPOS and STATPAK. Want to learn how to send a file to the printer? Or what to do when you see a 'recovery' message? No problem! We cover that too! In addition, there is an appendix summarizing many of the timesharing commands. Sound good? All this will be available (in about a week) for a very low price (CHEAP! CHEAP!) . See you in H-983 for your very own copy!

```
+-----+
|       The More the Merrier       |
+-----+
```

If you think the idea of a student-written timesharing manual is as bold and innovative as we do, then you'll love what is to come! The TXED manual is in its final stages of preparation and should be out very soon. It is a simplified yet fairly complete explanation of everything you will ever need to know about the TXED text editor but were unable to find in the Computer Centre manual. Also, there are several other manuals still in the planning stages.

```
+-----+
|   The Introductory Guide...   |
+-----+
```

The CUG Introductory Guide is a 20-page publication packed full of interesting information and humour on the subject of computing. It's a must for anyone who has anything to do with computing at Concordia... It's being typeset right now as I write this, and will be available by the end of January at our office in

room H-983. This momentous work will cost you -- are you ready for this? -- absolutely nothing! It's free! (So there's no excuse not to get one!)

```
+-----+
|   Real Programmers Don't...   |
+-----+
```

A classic article about computing appeared in the July 1983 issue of DATAMATION, entitled "Real Programmers Don't Use PASCAL". It's the sort of thing you absolutely have to read in order to appreciate... And thanks to CUG, you can read this fascinating story free of charge. Just log on to your favourite Cyber computer (825 or 835, it doesn't matter), and type GET,RP/UN=CCUSC51. Then you can either print it (using the command PRINT,RP,RS,CC) or look at it on the terminal (using the commands ASCII and COPY,RP). To get a print-out of it from JEF, just use the control cards GET,RP/UN=CCUSC51 and COPY,RP.

```
+-----+
|       It Had to Happen       |
+-----+
```

It's been said that every computer system has to have at least one picture of Snoopy on it somewhere... So now, courtesy of CUG, we have one here. On either Cyber, just GET,SNOOPY/UN=CCUSC51 and then COPY,SNOOPY (or, to print it on the line printer, PRINT,SNOOPY,RS). Enjoy...

```
+-----+
|       More Bad News          |
+-----+
```

Last November it was my sad duty to announce the resignations of Yvon Bourdeau and Lorna Fallavollita; this month I am equally saddened by the need to announce two more resignations from the CUG executive. Scott Maynard happily resigned when he received an offer of full time employment in Ottawa in his native field, civil engineering. All of us wish him the best of luck in his new job. Mark Zelada also resigned due to circumstances beyond his control.

We wish him the best. We, Steven, James, Lora, John and Micheline, wish our ex-execs the best of luck in their post-CUG adventures.

Of course you realize, this means by-elections!!!

+-----+
| A Tale of Two Passwords |
+-----+

Bear with me. If you don't have a Cyber account, skip to the next article.

Are you aware that you have two passwords for your account? There is a different, distinct password for each major subsystem to which you have access: BATCH and TIMESHARING. The batch password is valid for SUBMIT and ROUTE jobs while the timesharing password, for regular timesharing and JETS use. Does this mean that you will ever have to remember two different passwords? Of course not!

The fact is that both passwords start out the same when the Computer Centre assigns you your account; the trick is to keep them the same. It is quite simple: ALWAYS use the +PASSWOR command on timesharing. DO NOT use the PASSWOR command on timesharing! Remember '+PASSWOR'!!! DO NOT change your password on JETS!!! ALWAYS change your password on timesharing. Before using the +PASSWOR command, you must make sure that you are in the BATCH subsystem of timesharing. To ensure this, type 'BAT' as soon as you log on. The format for the command is +PASSWOR,oldpw,newpw where 'oldpw' and 'newpw' are your old and new passwords respectively. Using this command will change both your batch and timesharing passwords. Note that this assumes that both of them are the same to begin with. The PASSWOR command only changes your timesharing password. If you have any problems, go to room H-925 and ask for a Password Regeneration Form. It will take about 24 hours to get everything straightened out.

One word of caution about +PASSWOR: if your two passwords are

NOT the same for some reason, you will not be notified. It will happily go about changing your timesharing password to the new one, but will leave your batch password as it was before, not telling you of the problem.

One final thing, in the November issue there was an article about transferring files from one machine to the other. The password you use in the control cards is the BATCH password of the account you are sending the file to. This particular nuance has caused the PODs a great amount of hardship. Please make sure that you know what the batch password is, not just the timesharing one. If you are sending a file to a friend and you are having a lot of trouble, try his/her original password.

+-----+
| Engineering Week |
+-----+

As this is being written, Engineering Week is in full swing with everyone having a drunken good time at all the traditional events. However, there is a more serious side to Engineering Week. All week in the mezzanine, Engineering and Computer Science got a chance to show what they could do. It involved students showing what they can do within their field rather than within a bar. All the departments in Engineering and Computer Science were involved, each with a display of what the faculty and students are up to. CUG ran the display for Computer Science, which included the work of both professors and students. The faculty provided the most popular exhibit - a speech synthesizer. It provided hours of experimentation to many passers-by, each trying to get the machine to pronounce their name properly. Their iz a sirten trik too it.

Demonstrations provided by students included a program to simulate a programmer-on-duty, a program to make two IBM PCs send messages back and forth (thanks are due to the people at the CO-OP and the people at Phillips for the use

of two Micoms), also a program to select one of many computer games to play. The games themselves were mostly written by Concordia students. For much of the time, CUG showed its procedure file to all interested, encouraging them to make use of it.

```
+-----+
| Programming Hints -- Part 6 |
+-----+
```

I was planning to cover a lot of little hints this month to offset the detailed discussion of program design in the last couple of columns, but as you can see, that schmuck of an editor hasn't given me much of an opportunity. So, we'll all just have to content ourselves with a few pearls of wisdom. Why is he telling us all this...

WRITE THE PROGRAM TO WORK THE FIRST TIME: All the design wisdom in the world will not help you if you are a sloppy coder. Wirth is supposed to have said, regarding the actual coding of programs, that the devil is in the details. I think that this is largely true; the hardest part about programming is not writing the code, but writing good code. Good code works properly. A good coder never seems to spend all his/her sitting at a terminal, debugging the hell out of everything in sight, going cross-eyed reading debug traces.

The question is, how do you go about becoming proficient at coding? The answer is really quite simple: using the right approach and practicing it. There is no definitive 'right approach' valid for everyone, but there are a few general principles you should follow.

First of all, make sure you know EXACTLY what you are required to do. It is a very good idea to spend a lot of time working out a detailed algorithm. The more time you spend on your algorithm, the less time you will actually spend coding. When you sit down to start

writing, have either flowcharts or pseudocode at your side.

Next, choose the language most suitable for your task (assuming you are given a choice) taking into consideration how well you know the features that you are going to use. Once you decide on the implementation language, know all the relevant language details very well -- including syntax. Now you have the knowledge to begin coding in earnest.

The process of coding is long and complex, as is any activity that transforms an abstract idea into concrete terms. Start by writing the most abstract part of your program -- the main program. Given the inter-relationships amongst the modules of your program, you can go to the rest.

You should continue in a very structured way; do not be afraid to tackle the hard stuff first. After the main program, you might consider writing the I/O routines next, thus giving yourself a base to continue.

The key factor is to spend all your time writing the program and very little time debugging. For every hour you spend writing, you will save two hours of debugging; the trick is simple: write a program which is completely bug-free or at least find the bugs before you type them in. When you write your procedures on paper (NEVER type a program directly in at a terminal, this is the kiss of death), do not stop writing and revising the procedure until you are sure it will do what you want. The more procedures you write, the more you will revise old ones to get the interfaces correct. When you finally think the section of code you are working on is perfect, look at it again -- play computer.

For more info on design, see the last few issues. This method is not guaranteed to produce error-free programs, but it will cut down on time wasted.